



## COMPUTER SYSTEM

Publication number: JP2000330806

Publication date: 2000-11-30

Inventor: ENDO YOSHINORI; NAKAMURA KOZO; KAMIWAKI  
TADASHI; SAITO MASAHICO; ONO HIROSHI;  
NAKAMURA TOMOAKI

Applicant: HITACHI LTD

Classification:

- International: G06F13/10; G06F9/06; G06F9/455; G06F9/46;  
G06F9/48; G06F13/10; G06F9/06; G06F9/455;  
G06F9/46; (IPC1-7): G06F9/46; G06F9/06; G06F9/46;  
G06F13/10

- European: G06F9/48C2; G06F9/455H; G06F9/48C4

Application number: JP19990140914 19990521

Priority number(s): JP19990140914 19990521

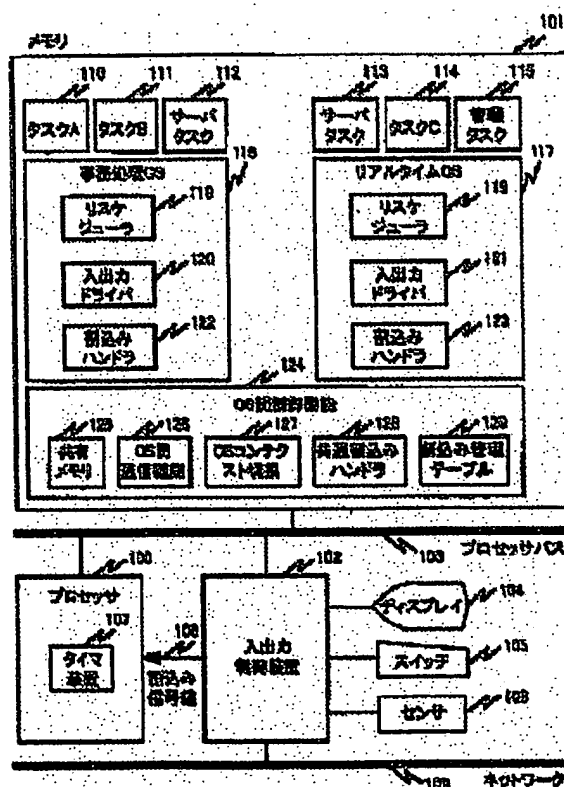
Also published as:

EP1054322 (A2)  
US6615303 (B1)  
EP1054322 (A3)

Report a data error here

## Abstract of JP2000330806

**PROBLEM TO BE SOLVED:** To bring out excellent merits from each of operating systems (OSs) and to reduce the cost of the systems in the case of sharing a peripheral device connected to a computer by plural OSs in a computer system capable of driving plural different OSs by a single processor. **SOLUTION:** In the computer system having plural OSs 116, 117, an OS context switch 127 for switching execution environments and a common interruption handler 128 for distributing a generated interruption to respective OSs 116, 117, the handler 128 has a means for referring to a priority interruption table for storing an OS to be started in each interruption factor, requesting the switch 127 to switch a current OS to an OS corresponding to an interruption factor and starting the interruption handler 122 or 123.



Data supplied from the esp@cenet database - Worldwide

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号  
特開2000-330806  
(P2000-330806A)

(43) 公開日 平成12年11月30日 (2000.11.30)

(51) Int.Cl. <sup>7</sup>		識別記号	F I		テーマコード(参考)
G 0 6 F	9/46	3 5 0	G 0 6 F	9/46	3 5 0 5 B 0 1 4
		3 4 0			3 4 0 A 5 B 0 7 6
	9/06	4 1 0		9/06	4 1 0 D 5 B 0 9 8
	13/10	3 3 0		13/10	3 3 0 A

審査請求 未請求 請求項の数7 OL (全 21 頁)

(21) 出願番号 特願平11-140914

(22) 出願日 平成11年5月21日 (1999.5.21)

(71) 出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(72) 発明者 遠藤 芳則

茨城県日立市大みか町七丁目1番1号 株式会社日立製作所日立研究所内

(72) 発明者 中村 浩三

茨城県日立市大みか町七丁目1番1号 株式会社日立製作所日立研究所内

(74) 代理人 100075098

弁理士 作田 康夫

最終頁に続く

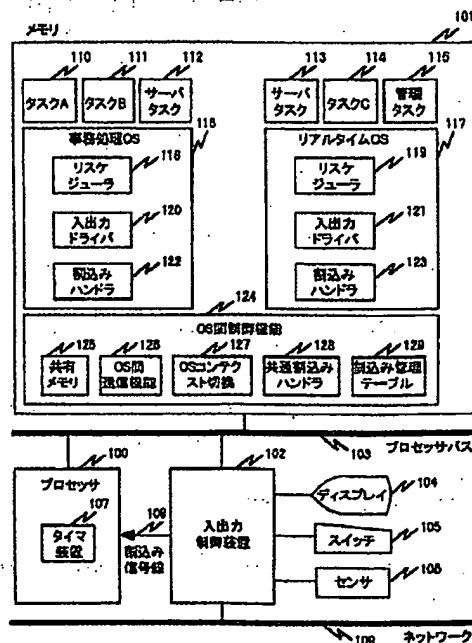
(54) 【発明の名称】 計算機システム

(57) 【要約】

【課題】 複数の相異なるオペレーティングシステムを単一のプロセッサで動作させる計算機システムにおいて、計算機に接続された周辺デバイスを該複数のオペレーティングシステムから共有するあたり、各オペレーティングシステムの優れた特長を引き出し、かつシステムコストを削減することを目的とする。

【解決手段】 複数のオペレーティングシステム 116、117 と、実行環境を切り替える OS コンテキスト切替 127 と、発生した割り込みを各オペレーティングシステムに振り分ける共通割り込みハンドラ 128 とを有する計算機システムにおいて、共通割り込みハンドラ 128 は割り込み要因毎に起動するオペレーティングシステムを記憶する優先割り込みテーブル 128 を参照し、割り込み要因に応じたオペレーティングシステムに切り替えるように OS コンテキスト切替 127 に依頼すると共に、割り込みハンドラ 122 ないし 123 を起動する手段を有する。

図 1



## 【特許請求の範囲】

【請求項1】複数のオペレーティングシステムと、複数のオペレーティングシステムを切り替えるOS切替手段とを有する計算機システムにおいて、  
 割込み要因毎に起動するオペレーティングシステムを記憶する優先割込みテーブルと、優先割込みテーブルを書き換える優先割込みテーブル書き換え手段とを備え、  
 OS切替手段は割込み要因から優先割込みテーブルを参照し対応するオペレーティングシステムに切り替えると共に、該オペレーティングシステムが備える割込み処理手段を呼び出すことを特徴とする計算機システム。

【請求項2】請求項1に記載の計算機システムにおいて、  
 優先割込みテーブルは、割込み要因毎に起動するオペレーティングシステムに加え、複数のオペレーティングシステムで共有する周辺デバイスからの割込みか否かを記憶することを特徴とする計算機システム。

【請求項3】請求項2に記載の計算機システムにおいて、  
 OS切替手段は、割込み要因から優先割込みテーブルを参照し、該割込み要求が複数のオペレーティングシステムで共有する割込みか否かを判定し、共有する割込みと判断したとき割込みを生成した周辺デバイスの内容を解析することで割込み対象とするオペレーティングシステムを判定し、オペレーティングシステムを切り替えると共に、該オペレーティングシステムが備える割込み処理手段を呼び出すことを特徴とする計算機システム。

【請求項4】複数のオペレーティングシステムと、複数のオペレーティングシステムを切り替えるOS切替手段とを有する計算機システムにおいて、  
 複数のオペレーティングシステムから共有する周辺デバイスと、前記共有する周辺デバイスとデータ入出力するデータ入出力サーバを該一つのオペレーティングシステムに備え、  
 前記オペレーティングシステム以外に備えられた前記共有する周辺デバイスとデータ入出力するデータ入出力クライアントは、データ入出力サーバにデータ入出力を依頼し、データ入出力サーバにおいて周辺デバイスとデータ入出力した結果を受信することでデータ入出力を実行することを特徴とする計算機システム。

【請求項5】請求項4に記載の計算機システムにおいて、  
 割込み要因毎に起動するオペレーティングシステムを記憶する優先割込みテーブルと、優先割込みテーブルを書き換える優先割込みテーブル書き換え手段とを備え、  
 OS切替手段は、割込み要因から優先割込みテーブルを参照し、該割込み要求が複数のオペレーティングシステムで共有する割込みかを判定し、共有する割込みと判断したとき該データ入出力サーバが備える割込み処理手段を呼び出すことを特徴とする計算機システム。

【請求項6】請求項4又は5に記載の計算機システムにおいて、  
 データ入出力サーバは最も優先順位の高いオペレーティングシステムが管理する領域に実装することを特徴とする計算機システム。

【請求項7】請求項4、5又は6に記載の計算機システムにおいて、  
 複数のオペレーティングシステムの間で互いに通信するOS間通信手段とを備え、  
 OS間通信手段を用いデータ入出力サーバとデータ入出力クライアントとの間で通信することを特徴とする計算機システム。

## 【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は複数のオペレーティングシステムを単一のプロセッサで動作させる計算機システムにおける周辺デバイスとのデータ入出力方法を対象とし、特に、複数のオペレーティングシステムが同一の周辺デバイスを共有に使用するための制御方法に関連する。

【0002】

【従来の技術】通常の計算機では、1つのオペレーティングシステムが動作し、それが計算機のプロセッサ、メモリ、および2次記憶装置といった計算機資源を管理し、計算機が効率よく動作できるように資源スケジュールを実施している。ところで、オペレーティングシステムには様々な種類がある。バッチ処理に優れるものや、事務処理といったGUIに優れるもの、実時間処理に優れているものなど様々である。これら複数の特徴を引き出すため、複数のオペレーティングシステムを1台の計算機で同時に実行したいというニーズがある。例えば、大型計算機においては、実際の業務に伴うオンライン処理を実行するオペレーティングシステムと、開発用のオペレーティングシステムを1台の計算機で動作させたいという要求がある。あるいはGUIが整っているオペレーティングシステムと実時間性に優れているオペレーティングシステムを同時に稼働させたいという要求もある。

【0003】1台の計算機上で複数のオペレーティングシステムを動作させる機構としては、大型計算機で実現されている仮想計算機方式(OSシリーズ第11巻、V.M.岡崎世雄著、共立出版株式会社)がある。仮想計算機方式では、仮想計算制御プログラムが全ハードウェアを占有して管理し、それを仮想化して仮想計算機を構成する。仮想計算機を構成する制御部は、物理メモリ、入出力機器装置、外部割り込み等を仮想化する。例えば、分割された物理メモリは、仮想計算機に対してはあたかも0番地から始まる物理メモリのように振る舞い、入出力装置を識別する装置番号も同様に仮想化されている。更に、磁気ディスクの記憶領域も分割して磁気ディスク

装置の仮想化まで実現している。

【0004】一方、1台の計算機で複数のオペレーティングシステムのインタフェースを提供する技術としてマイクロカーネルがある。マイクロカーネルでは、マイクロカーネルの上にユーザに見せるオペレーティングシステム機能を提供するオペレーティングシステムサーバを構築し、ユーザはそのサーバを経由して計算機資源を利用する。オペレーティングシステム毎のサーバを用意すれば、ユーザに様々なオペレーティングシステム環境を提供することができる。

【0005】

【発明が解決しようとする課題】例えば、車載ナビゲーションシステムでは、外部環境変化に対して即座にตอบสนองするというリアルタイム性と、いかなる場合でもシステムが停止しないという信頼性が重要視されている。このため、割込み応答性が高く、かつ、コンパクトでモジュール構成を有するリアルタイム用オペレーティングシステム（リアルタイムOS）が用いられることが多い。しかしながら、リアルタイムOSは、リアルタイム性・信頼性を重要視する反面、人間とのインタフェースに優れているとは言い難い。一方、一般のパソコン（PC）に用いられている事務処理用オペレーティングシステム（事務処理OS）は、GUIといった直接画面を操作する環境が準備され、対人間インタフェースに優れている。このため、従来はリアルタイムOSを使用していた分野にも事務処理OSのユーザインタフェースを使用したいという要求が高まっている。しかし、事務処理OSは人間との対話を主な処理としているため、割込み応答性よりも処理のスループットを重要視し、比較的長時間割込み禁止状態のまま処理を実行することもある。また、コンパクトな構成を有するリアルタイムOSに対し、信頼性において匹敵するとは言い難い。

【0006】しかしながら、大型計算機上で複数の仮想計算機（オペレーティングシステム）を並行して動作させる方式と同様に、組み込みシステムにおいて同一計算機システム上で事務処理OSとリアルタイムOSとを動作させ、必要に応じてこれらのオペレーティングシステムを切り替えることができれば、優れたユーザインタフェースとリアルタイム性・信頼性とを両立させることができる。マイクロプロセッサの性能向上を考慮すれば、複数のオペレーティングシステムを一つの計算機システム上で動作させることは、大型計算機だけに許された技術ではなくなってきている。

【0007】大型計算機ではマルチユーザに対応する必要があるが、組み込み機器では最低限シングルユーザに対応できればよい。そこで、リアルタイムOSと事務処理OSを一つの計算機システムで動作させるとき、それぞれのオペレーティングシステムの重要性を考慮し、リアルタイムOSが実行すべき処理がある場合はリアルタイムOSを優先的に動作させ、リアルタイムOS上で実

行するタスクがなくなったとき事務処理OSを実行するという最も単純なオペレーティングシステム切替え方式が適用可能である。このようなオペレーティングシステムの切替え方法では、周辺デバイスとの入出力において、入出力装置毎に割込み要求を受けるオペレーティングシステムを静的に決定し、発生した割込み番号を判定することでオペレーティングシステムを切り替える方法が用いられる。例えば、前記手法を車載ナビゲーションシステムに適用したとき、位置決定に必要なセンサからの割込みはリアルタイムOSの、人間とのインタフェースに用いるスイッチやディスプレイコントローラからの割込みは事務処理OSの割込みハンドラを起動するようにオペレーティングシステムを切り替える。しかしながら事務処理OSの信頼性は必ずしも高くないため、本適用例では事務処理OSの処理が停止すると同時にユーザインタフェースも使用不可能になり、車載ナビゲーションシステム全体の信頼性をも低下させてしまう。前記課題を解決するには、絶対に停止させたくない処理はリアルタイムOS側に、停止を許容する処理は事務処理OSにそれぞれ分離すればよいが、例えばユーザインタフェースのための入出力処理をリアルタイムOS側で処理してしまうと、ユーザインタフェースに優れた事務処理OSを導入した意味がなくなってしまう。よって、入出力デバイスを複数のオペレーティングシステム、即ち事務処理OSとリアルタイムOSの両方から使用できることが必須となり、上記説明した割込み番号から入出力処理を実行するオペレーティングシステムを静的に決定する方法では不十分であり、実用に耐えないという課題を有する。

【0008】本発明の目的は、複数の相異なるオペレーティングシステムを一つの計算機システム上で動作させるマルチオペレーティングシステム制御装置において、計算機に備えられた入出力デバイスを複数のオペレーティングシステムで共有する仕組みを提供することにある。

【0009】

【課題を解決するための手段】本発明では、上記課題を解決するため、第1の発明では、割込み要因毎に起動するオペレーティングシステムを記憶する優先割込みテーブルと、優先割込みテーブルを書き換える優先割込みテーブル書き換え手段とを備え、OS切換手段は割込み要因から優先割込みテーブルを参照し対応するオペレーティングシステムに切り替えると共に、該オペレーティングシステムが備える割込み処理手段を呼び出すことで計算機に備えられた入出力デバイスを複数のオペレーティングシステムで共有する。

【0010】第2の発明では、複数のオペレーティングシステムから共有する周辺デバイスと、前記共有する周辺デバイスとデータ入出力するデータ入出力サーバを該一つのオペレーティングシステムに備え、前記オペレー

ティングシステム以外に備えられた前記共有する周辺デバイスとデータ入出力するデータ入出力クライアントは、データ入出力サーバにデータ入出力を依頼し、データ入出力サーバが周辺デバイスとデータ入出力した結果を受信しデータ入出力処理することで計算機に備えられた入出力デバイスを複数のオペレーティングシステムで共有する。

【0011】

【発明の実施の形態】本発明の実施例を図面で説明する。

【0012】図1に本発明の第一の実施例における全体構成を示す。通常、計算機システムは、プロセッサ100、メモリ101、入出力制御装置102、ディスプレイ104、スイッチ105、センサ106などから構成される。プロセッサ100、メモリ101、入出力制御装置102はプロセッサバス103によって接続される。プロセッサ100は複数のオペレーティングシステムを動作させるためのマイクロプロセッサである。メモリ101は、オペレーティングシステム116、117、各オペレーティングシステム上で動作するタスクプログラム110～115、各オペレーティングシステムのための入出力ドライバプログラム120、121、割込みハンドラプログラム122、123、OS間制御機能プログラム124を保持する。これらのプログラムはプロセッサ100によって読み出されて実行される。

【0013】入出力制御装置102は、画面表示用デバイスであるディスプレイ104、ユーザからの指示を受けるスイッチ105、周囲環境の変化を測定するセンサ106等を接続する。また、工場／プラント制御用、あるいは、組込み向け計算機システムを実現する場合、入出力制御装置102にはネットワーク109が接続されることがある。ネットワーク109には、通信機器などの入出力機器が接続される。なお、入出力制御装置102に接続される入出力装置104～106のうち、いずれかまたは全ては、システム構成によって省略されることもあり、さらに多様な入出力装置が接続されることもある。入出力制御装置102は、割込み信号線108によってプロセッサ100と接続され、入出力動作完了などを通知する。図1では説明のため割込み信号線108とプロセッサバス103が別の装置であるように記載しているが、実際には、割込み信号線108はプロセッサバス103の一部である。プロセッサ100内部にはタイマ装置107が設けられており、一定周期毎に内部割込みを発生させる。タイマ装置107からの割込みは、オペレーティングシステムの計時などに使用される。

【0014】プロセッサ100は割込み信号線108によって通知される外部割込みやタイマ装置107などからの内部割込みをマスクできる機能を装備するものとする。割込みマスクとは、プログラムが割込みマスクを解除するまで、特定の割込みが入ることを遅延させる機能

である。一般に、割込みマスク機能には、下記の三種類が存在する。

【0015】(1) 全割込みマスク：全ての割込みをマスクする。

【0016】(2) 個別割込みマスク：個々の割込みをそれぞれマスクできるようにする。

(3) 割込みレベルマスク：各割込みにレベルを設定し、指定レベル以下の割込みをマスクする。

【0017】プロセッサ100の種類によって、上記

(1) (2)の組合せ、または(1) (3)の組合せのいずれかを装備することが多い。後者の組合せを装備するプロセッサを採用する場合、対応する入出力装置の重要性や最小応答時間に応じ割込みレベルを割り当てることになる。例えば、短い応答時間が要求されるネットワーク109からの割込みを、ディスプレイ104、スイッチ105および記憶装置106などからの割込みより高いレベルに設定する。

【0018】本実施例では、計算機システム内に二つのオペレーティングシステム116、117が存在する場合を説明する。オペレーティングシステム116、117は、各自に割り当てられたメモリ、及びプロセッサ資源を用い、タスク110～115を実行する。図1では、オペレーティングシステム数が2個、全タスク数が6個の例を示しているが、これらの数値よりも多い、または、少ないオペレーティングシステム、タスクを実装することも可能である。本実施例では、オペレーティングシステム数の動的な変更は想定しないが、各オペレーティングシステムが動的にタスク生成・削除を行うことは可能である。さらに本実施例では、オペレーティングシステム116が事務処理OS、オペレーティングシステム117がリアルタイムOSであると仮定するが、実際に、各オペレーティングシステムがいかなる種類であっても、本発明で述べる技術は適用可能である。タスク110～112は事務処理OSで実行される事務処理タスクであり、タスク113～115はリアルタイムOSで実行されるリアルタイムタスクである。さらに、リアルタイムOSでの応答時間を保証するため、事務処理OSよりリアルタイムOSの割込み優先順位が高いと仮定し以下説明する。前記仮定では、リアルタイムOSのいずれかのタスクが実行状態の場合はリアルタイムOS 116がプロセッサ資源が使用し、すべてのタスクがアイドル状態またはウェイト状態の場合は事務処理OSにコンテキストが切り替えプロセッサ資源が使用する。

【0019】事務処理OS 116及びリアルタイムOS 117を一つのプロセッサ上で連携しながら動作させるため、OS間制御機能124を備える。OS間制御機能124は、事務処理OSとリアルタイムOS間の両者からアクセス可能な共有メモリ125、事務処理OSとリアルタイムOS間でメッセージ転送を行うOS間通信機能126、事務処理OSとリアルタイムOS実行環境を切

り替えるOSコンテキスト切替127、発生した割込みを各オペレーティングシステムに振り分ける共通割込みハンドラ128、割込み要求に対応するオペレーティングシステムや割込み開始アドレスを記憶する優先管理テーブル129とを備える。

【0020】共有メモリ125は、オペレーティングシステム間で高速なデータ交換を実現することを目的とし、事務処理OSとリアルタイムOSの両者から読み書きが可能である。ここで、データ競合を防止するためセマフォ機能を用い排他的に共有メモリをアクセスすることが望ましい。OS間通信機能126は、各オペレーティングシステムに対応するメッセージキューを準備し、各オペレーティングシステム間でメッセージを受け渡す。共通割込みハンドラ128は、割込み要求が発生したとき最初に呼び出され、優先割込みテーブル129に記憶された割込み番号毎のオペレーティングシステムを判定し、対応する割込みハンドラ122、123を呼び出す。OSコンテキスト切替127は、割込み要求やOS間制御機能の呼び出しによりオペレーティングシステムの切替が必要と判断されたときコンテキストを切り替える。割込み管理テーブル129は、割込み番号毎に呼び出される割込みハンドラが所属するオペレーティングシステム情報や、各オペレーティングシステム毎の割込みハンドラの開始アドレス情報を保持する。従来の方法では、割込み管理テーブル129をシステム起動時に静的に決定していたため、システム動作中にその内容を書き換えることはできなかった。本システムでは、システム動作中でも割込み管理テーブル129を書き換える手段を提供することで、一つの入出力装置を複数のオペレーティングシステムが共用することを可能としたことを特徴とする。

【0021】オペレーティングシステム116、117は、入出力装置との間でデータ入出力を処理する入出力ドライバ120、121、入出力装置からの割込み要求を受ける割込みハンドラ122、123を備える。割込みハンドラ122、123は、共通割込みハンドラ128から呼び出され、ユーザが定義した割込み処理プログラムを実行する。入出力ドライバ120、121は、各タスクから入出力装置を制御するためのインタフェースを提供し、入出力装置を読み書きすることでデータ入出力したり、入出力装置を制御するといった機能を提供する。リスケジューラ118、119は、タスクの生成・削除・停止・再開や、外部割込み・内部割込みに伴ってタスク切り替えを行わなければならない場合に起動される。リスケジューラは直前に実行していたタスクの実行環境（プログラムカウンタ、ステータスレジスタ、汎用レジスタなど）をタスク管理テーブルに格納し、新たに実行すべきタスクを決定し、そのタスクの実行環境をタスク管理テーブルから取り出し各レジスタに設定することで、選択したタスクを実行する。

【0022】図2に、本発明が前提とするプロセッサ100の内部構成例を示す。キャッシュメモリ130はメモリ101上のデータまたは命令を一時的に格納するバッファ記憶装置である。CPU131は演算回路であり、メモリ101若しくはキャッシュメモリ130上に存在する命令を順次読み出して実行する。命令実行に際して、演算結果を一時的に保持するための汎用レジスタ132、実行命令アドレスを保持するプログラムカウンタ133、実行状態を保持するステータスレジスタ134を用いる。キャッシュメモリ130、CPU131、汎用レジスタ132、プログラムカウンタ133、ステータスレジスタ134は、互いに、データ転送を行うデータバス135、アドレス指定を行うアドレスバス136によって接続されている。

【0023】割込み信号線108とタイマ装置107は割込みコントローラ137に接続される。割込みコントローラ137は、CPU130に対して割込み状態信号138を生成する役割を有する。割込み状態信号138は、プロセッサ100に対して現在如何なる種類の割込みが発生しているかを示す信号線である。通常、ステータスレジスタ134は現在の割込みマスクに関する情報を有しており、割込み状態信号138で指定される割込みを受け付けるか否かを決定する。割込みを受け付ける場合、割込みコントローラ137は、プログラムカウンタ133、ステータスレジスタ134などの値を書き換え、対応する割込み処理プログラムを実行させる。

【0024】ステータスレジスタ134の構成例を図3に示す。ここでは、プロセッサ100が全割込みマスク機能と割込みレベルマスク機能とを装備している場合の例を示す。図3の例では、ステータスレジスタ134が割込みブロックビット140と割込みマスクレベルフィールド141を有する。割込みブロックビット140がONである場合、プロセッサ100に対する全ての割込みがマスクされる。割込みマスクレベルフィールド141は現在の割込みマスクレベル値を示し、これ以下の割込みレベルは受け付けられない。図3の例では、割込みマスクレベルフィールド141は4ビット長である。このため、合計16種類のマスクレベルを指定可能であるが（通常、割込みレベル0は「割込みが発生していない」ことを意味し、割込みマスクレベルを0とすることは「割込みマスクを行わない」ことを意味するため、実質的には15種類となる）、割込みマスクレベルフィールド141のビット数を変更することにより、受理できる割込みレベルの種類を増減させることができる。

【0025】図4は、プロセッサ100が全割込みマスク機能と個別割込みマスク機能とを装備している場合のステータスレジスタ134である。この例では、ステータスレジスタ134は実際には2つのレジスタ（実行状態レジスタ142と割込みマスクレジスタ143）から構成される。図3と同様、実行状態レジスタ142内に

割込みブロックビット140が設けられている。割込みマスクレジスタ143内の割込みマスクビット144～147はそれぞれ別々の割込みに対応しており、該割込みマスクビットのいずれかをONとした場合、対応する割込みが受け付けられなくなる。図3に示すステータスレジスタの例は、図4のステータスレジスタの特殊例である。例えば、割込みマスクビット144のみONとなっている状態をレベル1とし、割込みマスクビット144、145の二つがONとなっている状態をレベル2、割込みマスクビット144～146の三つがONとなっ

ている状態をレベル3、…、のように対応させることができる。このため、以降、ステータスレジスタ134は図4に示す構成を有するとして説明する。  
【0026】一般的なプロセッサでは、割込みを受理すると、ハードウェアによって自動的に割込みブロックビット140がONに書き換えられる。必要に応じて、割込み処理プログラムが割込みブロックビット140をOFFに書き換えて割込み受け付けを許可することができる。また、オペレーティングシステムおよびタスクが、一時的に割込みブロックビット140や割込みマスクレジスタ143の内容を書き換え、特定割込みの受け付けを待たせることもできる。割込みマスク機能は、排他制御の実現や、割込み処理実行中に再び同一割込みが発生することを避けるために用いられる。

【0027】図5に割込みハンドラ122、123、共通割込みハンドラ128、OSコンテキスト切替127、割込み管理テーブル129の詳細を示す。

【0028】割込みハンドラ122、123は、割込み発生時のレジスタなどの保存、一時変数の保持を行うための領域として、それぞれ、割込みスタック151、153を所有する。割込み発生時には、割込み発生直前のレジスタ値を保存し、割込み処理終了後にはこのレジスタ値を元に復帰するため、割込みスタック151、153を備える。使用するプロセッサ100の種類によっては、割込み発生時に自動的にレジスタが切り替わり、かつ、割込み処理終了後に切替え前のレジスタに戻す機能を備えているものもある。しかしながら、多重割込みが可能なシステムを考慮した場合には、このようなハードウェアを使用していても、割込みスタックが必要となる（割込み処理中により高い緊急性を有する割込みが発生した場合、新たに発生した割込み処理を優先的に実行し、次いで、元の割込み処理に戻る必要がある）。割込みハンドラ122、123は、割込みスタック151、153において、どの領域まで使用したかを示すためのポイントとして、割込みスタックポイント150、152を所有する。割込みハンドラ122、123は、割込みスタック、および、割込みスタックポイントを用いて割込み発生前の実行環境（レジスタなど）を保存し、必要な割込み処理を実行する。割込み処理終了後、割込み発生前の実行環境を復帰して元々動作していたプログラ

ムの実行を継続するように動作する。

【0029】共通割込みハンドラ128は、発生した割込みを前記割込みハンドラ122、123に対して振り分ける役割を有する。このため、共通割込みハンドラ128は、現在実行しているオペレーティングシステムが事務処理OS116であるかリアルタイムOS117であるかを記憶する領域として、実行OS記憶変数154を所有する。例えば、図5に示した時点で事務処理OSが処理を実行しているものと仮定すると、この場合、実行OS記憶変数154には、「事務処理OS」が記憶されている。勿論、実行OS記憶変数154が文字列「事務処理OS」を記憶することは非常に非効率的であるため、例えば、事務処理OSであれば0、リアルタイムOSであれば1といった整数形式で記憶するとよい。

【0030】割込み管理テーブル129は、発生した割込みがいずれのオペレーティングシステムの割込みハンドラを起動するかを示す優先割込みテーブル156、および各オペレーティングシステムが備える割込みハンドラの開始アドレスを示す割込みアドレステーブル155を有する。優先割込みテーブル156は、例えば図7に示したような個々の割込みがどちらのオペレーティングシステムで処理されるべきかを示す対応表で構成され、フラグ1が立っているオペレーティングシステムが備える割込みハンドラに処理を依頼する。さらに、複数のオペレーティングシステムで共有する入出力装置からの割込みにはフラグ1を、共有しない割込みにはフラグ0を記憶する。本実施例では、16種類の割込み要因が存在すると仮定すると、割込み対応テーブル156も16個のエントリ（IRQ#0～IRQ#15）の構成になる。なお、本説明では優先ドライバテーブル160にフラグを記録する方法を想定したが、何らかの識別可能な情報を記録すればいかなる方法でも実現可能である。割込みアドレステーブル155は、例えば図6に示すような各々のオペレーティングシステムが備える割込みハンドラの開始アドレスを保持する表である。

【0031】本発明では、一つの入出力機器を複数のオペレーティングシステムから共用するという目的を達成するため、共通割込みハンドラ128が受けた割込み要求から起動するオペレーティングシステムを動的に変更する仕組みを提供する。優先割込みテーブル156の内容は、ユーザオペレーションや入出力機器の使用状況によって更新され、共通割込みハンドラ128は優先割込みテーブル156の内容に従って割込みハンドラを呼び出すように動作する。さらに詳細を説明すると、共通割込みハンドラ128は、実行OS記憶変数154を参照し、現在実行しているオペレーティングシステムの種類を判定する。オペレーティングシステムの種類が優先割込みテーブル156から得られた振り分け先オペレーティングシステムに一致しなければ、OSコンテキスト切替モジュール127にオペレーティングシステム切替



えを依頼する。前記オペレーティングシステム切替えが終了した場合、ないしオペレーティングシステムが一致し切り替えが不要な場合は、該当するオペレーティングシステムの割込みハンドラに処理を依頼する。なお、実行OS記憶変数154はOSコンテキスト切替モジュール127からも参照される。このように、OS間制御機能プログラム内の各モジュールが有する内部構造は、個々のモジュールによって占有されるものとは限らず、各モジュールが必要に応じて共用できるものとする。

【0032】OSコンテキスト切替モジュール127は、OS間制御機能の呼び出しや割込み発生の結果、オペレーティングシステムを切り替えなければならない場合に起動される。二つのオペレーティングシステムを切り替えるため、これらの実行環境（すなわち、レジスタ値）を保存しておくための領域を所有する。図中では、事務処理OSの実行環境を保存するための領域として、事務処理OS用保存コンテキスト157、リアルタイムOSの実行環境を保存するための領域として、リアルタイムOS用保存コンテキスト158を用意する。OSコンテキスト切替モジュール127は、コンテキストを切り替える際に現在実行しているオペレーティングシステムに対応する保存コンテキストに実行環境を保存し、次に、もう一方の保存コンテキストから実行環境を読み出して、プロセッサ100のレジスタに設定する。これにより、オペレーティングシステムの切替えを実施できる。以降、図5で示したモジュールのうち、OSコンテキスト切替モジュール127、共通割込みハンドラ128、割込みハンドラ122の処理フローの説明を行う。割込みハンドラ123に関しては、割込みハンドラ122と同様の処理フローを有するため、説明を省略する。

【0033】図8は、OSコンテキスト切替モジュール127の処理フローである。OSコンテキスト切替モジュール127は、実行するオペレーティングシステムを切り替えなければならない場合にのみ呼び出され、切替えが必要か否かのチェックは呼出し前に行われている。従って、実行OS記憶変数154を参照し、どちらのオペレーティングシステムへ切り替えなければならないかをチェックする（処理160）。ここで、事務処理OS116からリアルタイムOS117への切替えを行う場合、事務処理OS用保存コンテキスト157内に使用中レジスタの値を退避する（処理161）。次に、リアルタイムOS用保存コンテキスト158から実行環境を復帰して、レジスタに設定することにより（処理162）、以前退避した状態からリアルタイムOS117を再実行させることができる。リアルタイムOS117から事務処理OS116への切替えを行う場合、逆に、リアルタイムOS用保存コンテキスト158に使用中レジスタの値を退避し（処理163）、次に、事務処理OS用保存コンテキスト157からレジスタを復帰する（処理164）。いずれの場合においても、最後に、切替え後

のオペレーティングシステムの種類を実行OS記憶変数154に書き込む（処理165）。

【0034】図9は、共通割込みハンドラ128の処理フローを示している。一般に、単一のオペレーティングシステムによって制御される計算機システムでは、一旦、全ての割込みを割込みハンドラと呼ばれるモジュールが処理し、次いで、各プログラムに振り分ける。しかしながら、本実施例で説明するような複数個のオペレーティングシステムを動作させる計算機システムでは、この更に前に、共通割込みハンドラが全ての割込みを受け、これに対応するオペレーティングシステムの割込みハンドラに振り分ける形となる。各オペレーティングシステムに割込みを振り分けるに当たって、割込み発生時に、割込み対象とは別のオペレーティングシステムが処理を実行していることもある。この場合、割込みに対応するオペレーティングシステムに切り替える必要があり、共通割込みハンドラはこのような役割も有することになる。

【0035】共通割込みハンドラ128は、割込み発生後、第一に、実行OS記憶変数154の内容を取り出して、現在実行中のオペレーティングシステムが事務処理OS116であるかリアルタイムOS117であるかをチェックする（処理170）。次に、優先割込みテーブル156を参照し、発生した割込み要求がいずれのオペレーティングシステムの割込みハンドラを起動すべきかを入手する（処理171）。図6に示す優先割込みテーブル156に記載された値を用いた場合を想定すると、割込みIRQ#0が発生した場合にはリアルタイムOS117、割込みIRQ#1が発生した場合には事務処理OS116、…、割込みIRQ#15が発生した場合にはリアルタイムOS117、というように対応するオペレーティングシステムを入手することができる。次に、得られた割込み対象オペレーティングシステムが実行中オペレーティングシステムに等しいか否かをチェックする（処理172）。割込み要求が現在実行中のオペレーティングシステムに対するものでなければ、一旦、オペレーティングシステムの切替えを行わなければならない。この切替え処理は、OSコンテキスト切替モジュール127に依頼することによって行う（処理173）。次に割込み対象オペレーティングシステムが事務処理OS116であるかリアルタイムOS117であるかをチェックし（処理174）、対象が事務処理OS116であれば、割込みアドレステーブル155を参照し事務処理OSの割込みハンドラ122を起動する（処理175）。リアルタイムOS117への割込みが発生していれば、同様に割込みアドレステーブル155を参照しリアルタイムOSの割込みハンドラ123を起動することになる（処理176）。一般に、割込みハンドラ122、123は、割込み処理でタスク切替えを行わなければならないとき、リスケジューラ118、119を呼び出し、共通割込みハ

ンドラ128に制御を戻すことはない。しかし、タスク切替えが発生しない場合、そのまま割込みハンドラの処理を終了する。このとき、割込みハンドラ122、123は、共通割込みハンドラへ制御を戻し(図10で後述)、共通割込みハンドラは処理177から動作を再開する。処理177は、割込み発生時にオペレーティングシステムを切り替えたかどうかをチェックする処理である。処理173によってオペレーティングシステムを切り替えた場合、ここで再び、オペレーティングシステムを切り替え、元の実行環境に戻す必要がある。このため、もう一度OSコンテキスト切替モジュール127に依頼して、切替え処理を実行させる(処理178)。

【0036】図10は共通割込みハンドラ128から呼び出される事務処理OSの割込みハンドラ122の処理フローを示したものである。第一に、使用中レジスタを割込みスタック151に退避し(処理180)、割込み処理を実行する(処理181)。ここで、オペレーティングシステムがリスケジューリング中か否か(すなわち、リスケジューラ118の処理実行中か否か)をチェックする(処理182)。リスケジューリング中とは、次に実行すべきタスクを選択している最中であり、実際にはタスクを実行していないことを意味する。したがって、オペレーティングシステムの処理を単純化させるならば、この場合、再びリスケジューリングを最初からやり直せば良いことになる。すなわち、リスケジューリング中であると判定した場合、割込みスタックの退避レジスタを破棄し(処理187)、リスケジューラ118を最初から起動する(処理188)。なお、リスケジューリング中に割込みが発生しても、実行すべきタスクを新たに選択し直す必要がない場合もある。本方式は、この状況でもリスケジューリングを最初から行うことになり、効率的でないこともある。このため、リスケジューリング中に発生し、リスケジューラ118の実行に影響を与える処理(例えば、タスク起動・終了など)を待ち行列などに登録しておく方法を用いることもできる。この場合、リスケジューラ118の終了前で該待ち行列に登録しておいた処理を纏めて実行する。この方式を採用すると、割込み発生のたびに途中まで実行していたリスケジューリングをやり直す必要が無い。しかしながら、当該待ち行列に登録していた処理を纏めて実行した後、これによってリスケジューリングが再び必要となるか否かを再チェックしなければならない。なお、本実施例では、単純化のため、前者の方式で説明を行う。しかし、後者の方式を用いる場合、リスケジューリング中か否かを示すフラグ、ないし処理待ち行列を用意すればよい。オペレーティングシステムが提供するシステムコールなどにおいて、リスケジューラ118の実行に影響を与える処理を、リスケジューリング中であれば、該待ち行列に登録する。更に、リスケジューラ118の処理フロー中に、該待ち行列に登録された処理を一括して実行するモ

ジュールを挿入する。

【0037】割込みハンドラ122がリスケジューリング中でないと判定した場合、当該オペレーティングシステムはその時点で何らかのタスクを実行している。このため、第一に、タスク切替えが必要か否かを判定する(処理183)。タスク切替えが必要なければ(現実行タスクが最高優先順位であり、かつ、実行可能であれば)、このまま、割込みハンドラ122の処理を終了する。このため、割込みスタック151上に退避しておいたレジスタ値を復帰し(処理184)、共通割込みハンドラへ制御を戻す(処理185)。タスク切替えが必要であると判定した場合、割込みスタック151上に退避したレジスタの値をタスク管理テーブルにコピーし(処理186)、割込みスタック上の退避レジスタを破棄する(処理187)。この後、リスケジューラ118を起動する(処理188)。なお、プロセッサ100に、メモリ101上のデータの参照を行うと共に割込みスタックポインタ150の値を増減できる機能が備わっていれば、処理186と処理187を纏めて実行することができる。

【0038】以上、OS間制御機能124に備えられた優先割込みテーブル156を参照し、動作させるオペレーティングシステムを決定した後、割込みハンドラを起動することで、入出力装置を複数のオペレーティングシステムで共用する方法を説明したが、入出力装置によっては割込み番号から一意に動作させるオペレーティングシステムを決定できない場合も発生する。例えば図11に示す車載ナビゲーションシステムの入出力パネルには、ユーザが指示を入力するためのスイッチ105と、結果を表示するディスプレイ104とを備える。スイッチ105は、それぞれのリアルタイムOS117に備えられたタスクのみが使用するリアルタイムOS専用スイッチ190、事務処理専用スイッチ191、双方のオペレーティングシステムに備えられたタスクから共用可能な共用スイッチ192を備えることが望ましい。このように、オペレーティングシステム毎に専用スイッチを設けることで、頻繁に使用する機能を固定のスイッチに割り当てることが可能になり、操作性が向上するという特徴が得られる。さらに固定のスイッチに機能を割り当てることで、ボタンに機能名称を表示可能になり識別性が向上する。しかし、これらスイッチ毎に異なる割込み番号を付与することは、割込み信号線数の制約やハードウェアの簡略化の観点より不利である。一方、スイッチに同一の割込み番号を付与すると、スイッチが押下されたときいずれのオペレーティングシステムが備える割込みハンドラを起動すべきか判定できない。この問題を解決するため、OS間制御機能124に備えられた共有メモリ125に、スイッチテーブル200を設ける。スイッチテーブルには、各スイッチ毎にいずれのオペレーティングシステムで使用するスイッチかの情報を記憶する。図

12では、スイッチテーブルには「リアルタイムOS」、「事務処理OS」、「共用」が記憶される。勿論、テーブルに文字列を記憶することは非常に非効率的であるため、例えばリアルタイムOSであれば0、事務処理OSであれば1、共用であれば2といった正数形式で情報を記憶するとよい。

【0039】次に、上記スイッチテーブル200を用い複数のオペレーティングシステムで入出力装置を共有する共通割込みハンドラ128の処理フローを図13を用い説明する。なお、ここで説明する処理はスイッチのみならず、割込みの内容を判断することで起動するオペレーティングシステムを判定することが必要な他の入出力装置に適用することも可能である。

【0040】共通割込みハンドラ128は、割込み発生後、実行OS記憶変数154の内容を参照し、現在実行中のオペレーティングシステムが事務処理OS116であるかリアルタイムOS117であるかをチェックする(処理210)。次に、優先割込みテーブル156を参照し、共有フラグが立っていない場合は、優先割込みテーブル156を参照し、発生した割込みがいずれのオペレーティングシステムに対応しているかを入手する(処理212)。共有フラグが立っている場合は、入出力装置にアクセスし割込み内容を判定することで割込み対象オペレーティングシステムを判定する(処理213)。ここでは、割込み内容とスイッチテーブル200を比較し、リアルタイムOS専用スイッチが押されたと判定した場合はリアルタイムOS117を、事務処理OS専用スイッチが押されたと判定した場合は事務処理OS116を割込み対象OSとしてそれぞれ記憶する。また共用スイッチが押されたと判定した場合は、優先割込みテーブル156を参照し、発生した割込みがいずれのオペレーティングシステムに対応しているかを入手する。ここで図7の優先割込みテーブル156と、図12のスイッチテーブル200に記憶された値を用い処理した場合を想定すると、スイッチ割込みにIRQ#2が割り当てられていれば、スイッチ#0が入力された場合はリアルタイムOS、スイッチ#2が入力された場合は事務処理OS、スイッチ#3が入力された場合は共用なので再度優先割込みテーブル156を参照しリアルタイムOSというように対応する割込み対象オペレーティングシステムを入手する。

【0041】次に、割込み対象オペレーティングシステムが実行中オペレーティングシステムに等しいか否かをチェックする(処理214)。発生した割込みが現在実行中のオペレーティングシステムに対するものでなければ、一旦、オペレーティングシステムの切替えを行わなければならない。この切替え処理は、OSコンテキスト切替モジュール127に依頼することによって行う(処理215)。次に割込み対象オペレーティングシステムが事務処理OS116であるかリアルタイムOS117

であるかをチェックし(処理216)、対象が事務処理OS116であれば、割込みアドレステーブル155を参照し事務処理OSの割込みハンドラ122を起動する(処理217)。リアルタイムOS117への割込みが発生していれば、同様に割込みアドレステーブル155を参照しリアルタイムOSの割込みハンドラ123を起動することになる(処理218)。一般に、割込みハンドラ122、123は、割込み処理によってタスク切替えを行わなければならないとき、リスケジューラ118、119を呼び出し、共通割込みハンドラ128に制御を戻すことはない。しかし、タスク切替えが発生しない場合、そのまま割込みハンドラの処理を終了する。このとき、割込みハンドラ122、123は、共通割込みハンドラ128へ制御を戻し、共通割込みハンドラ128は処理219から動作を再開する。処理219は、割込み発生時にオペレーティングシステムを切り替えたかどうかをチェックする処理である。処理215によってオペレーティングシステムを切り替えた場合、ここで再び、オペレーティングシステムを切り替え、元の実行環境に戻す必要がある。このため、もう一度OSコンテキスト切替モジュール127に依頼して、切替え処理を実行させる(処理220)。以上、OS間制御機能124に備えられた共通割込みハンドラから割込み管理テーブル129に記憶された優先割込みテーブル156を参照し、複数のオペレーティングシステムを1つの計算機で動作させるシステムにおいて入出力装置を共有する方法を説明したが、前記説明した方法で全てのケースに対応することはできない。例えば、グラフィックスアクセラレータやシリアルコミュニケーションといった入出力装置では、入出力ドライバから動作条件や各種動作パラメータを設定し、ハードウェア内部のレジスタで前記パラメータや動作状態を保持する。このような入出力装置に対し単純に前記処理を適用すると、各々のオペレーティングシステム内の入出力ドライバ、及び割込みハンドラは、オペレーティングシステムが切り替わったことを認識することができないため、以前のオペレーティングシステムの入出力ドライバが設定した状態から処理を継続しようとする。しかし、設定しようとするパラメータは、自オペレーティングシステムの入出力ドライバが最後に設定した値から継続したときのみ正常に動作し、オペレーティングシステムが切り替えられる前に何らかのパラメータが入出力装置に書き込まれていれば希望に沿った動作をしないのは明確である。

【0042】これら問題を解決するため、本発明では図14に示すように各々のオペレーティングシステムに備えられた入出力ドライバ120、121、及び割込みハンドラ122、123といった入出力処理にクライアント-サーバモデルを適用したことを特徴とする。即ち、複数のオペレーティングシステムの入出力処理の一つでサーバ側入出力処理222を実行し、その他のオペレーテ

イングシステムの入出力処理でクライアント側入出力処理 221 を実行する。サーバ側入出力処理 222 は、複数のオペレーティングシステムで共有する入出力装置 226 を管理するように動作し、入出力ドライバ 121 から入出力装置 226 へのデータ入出力 225 を実行したり、入出力装置 226 からの割込み要求を割込みハンドラ 123 で受けるように動作する。さらに、クライアント側入出力処理 221 からの依頼通知 223 を受け、その内容を判断し入出力装置 226 とデータ入出力 225 し、その結果を結果通知 224 としてクライアント側入出力装置 221 に返信するように動作する。一方、クライアント側入出力処理 221 は、入出力装置 226 へのデータ入出力をサーバ側入出力装置 222 に依頼通知 223 を送信し、返信された結果通知 224 を用い入出力処理を実行する。従って、クライアント側入出力処理 221 から入出力装置 226 のハードウェアを直接制御することではなく、常にサーバ側入出力装置が入出力装置を管理するため、動作状態に矛盾は生じない。

【0043】なお、サーバ側入出力処理 222 は優先順位の高いオペレーティングシステムに設定することが望ましい。本実施例では事務処理 OS よりリアルタイム OS の優先順位が高いと仮定しているため、サーバ側入出力処理 222 をリアルタイム OS に備える。優先順位の高いオペレーティングシステムにサーバ側入出力処理 222 を備えることで、割込み要求といった処理がマスクされることを防止できる。従って、割込み要求発生での応答時間が不当に遅くなることを防ぎ、データの取りこぼしを無くすることができるという特徴が得られる。

【0044】以降、図 14 で示したモジュールのうち、共通割込みハンドラ 128、割込みハンドラ 122、123 の処理フローの説明を行う。

【0045】図 14 に示すクライアント-サーバモデルを入出力処理に適用したとき、共通割込みハンドラ 128 の動作フローを図 15 を用い説明する。なお、基本的なシステム構成は図 1、及び図 5 と同様であり、さらに優先割込みテーブル 156 では、複数のオペレーティングシステムで共有する入出力装置の割込み番号には共有フラグ 1 が、特定のオペレーティングシステムが使用する入出力装置の割込み番号には 0 がそれぞれ記憶されているものとする。さらに、図 14 に示したように、サーバ側入出力処理 222 をリアルタイム OS 117 に、クライアント側入出力処理を事務処理 OS にそれぞれ備えると仮定する。

【0046】共通割込みハンドラ 128 は、割込み発生後、第一に、実行 OS 記憶変数 154 の内容を取り出して、現在実行中のオペレーティングシステムが事務処理 OS 116 であるかリアルタイム OS 117 であるかをチェックする (処理 230)。次に、優先割込みテーブル 156 を参照し (処理 231)、共有フラグが立っていない場合は優先割込みテーブル 156 を参照し、発生した

割込みがいずれのオペレーティングシステムに対応しているかを入手する (処理 232)。共有フラグ 1 が立っている場合は、リアルタイム OS を割込み対象オペレーティングシステムと判定する (処理 233)。

【0047】次に、割込み対象オペレーティングシステムが実行中オペレーティングシステムに等しいか否かをチェックする (処理 234)。発生した割込みが現在実行中のオペレーティングシステムに対するものでなければ、一旦、オペレーティングシステムの切替えを行わなければならない。この切替え処理は、OS コンテキスト切替モジュール 127 に依頼することによって行う (処理 235)。次に割込み対象オペレーティングシステムが事務処理 OS 116 であるかリアルタイム OS 117 であるかをチェックし (処理 236)、対象が事務処理 OS 116 であれば、割込みアドレステーブル 155 を参照し事務処理 OS の割込みハンドラ 122 を起動する (処理 237)。リアルタイム OS 117 への割込みが発生していれば、同様に割込みアドレステーブル 155 を参照しリアルタイム OS の割込みハンドラ 123 を起動することになる (処理 238)。一般に、割込みハンドラ 122、123 は、割込み処理によってタスク切替えを行わなければならないとき、リスケジューラ 118、119 を呼び出し、共通割込みハンドラ 128 に制御を戻すことはない。しかし、タスク切替えが発生しない場合、そのまま割込みハンドラの処理を終了する。このとき、割込みハンドラ 122、123 は、共通割込みハンドラへ制御を戻し、共通割込みハンドラは処理 239 から動作を再開する。処理 239 は、割込み発生時にオペレーティングシステムを切り替えたかどうかをチェックする処理である。処理 235 によってオペレーティングシステムを切り替えた場合、ここで再び、オペレーティングシステムを切り替え、元の実行環境に戻す必要がある。このため、もう一度 OS コンテキスト切替モジュール 127 に依頼して、切替え処理を実行させる (処理 240)。

【0048】図 16 は共通割込みハンドラ 128 から呼び出されるサーバ側入出力装置 222 の一部であるリアルタイム OS の割込みハンドラ 123 の処理フローを示したものである。

【0049】第一に、使用中レジスタを割込みスタック 153 に退避し (処理 250)、優先割込みテーブル 156 を参照することで割込み要求した入出力装置が複数のオペレーティングシステムから共有されているか判定する (処理 251)。共有していない場合は、リアルタイム OS に対する割込み要求であり、対応する割込み処理を実行する (処理 253)。共有している場合は、再度優先割込みテーブル 156 を参照し、発生した割込み要求がいずれのオペレーティングシステムに対応しているかを判定する (処理 252)。リアルタイム OS 側のフラグが立っている場合、ないし共有フラグが立っ

ていない場合は、リアルタイムOS割込みに対する割込み処理を実行する(処理253)。事務処理OS割込み側のフラグが立っている場合は、事務処理OS割込みに対する割込み処理を実施する(処理254)。ここで、事務処理OS割込みに対する割込み処理254は、割込みが発生したことをクライアント側の割込みハンドラ122に通知するため、事務処理OS116へのソフトウェア割込みを生成する。さらに必要があれば前記ソフトウェア割込み処理で入手した情報を、共有メモリ125に書き込む。なお生成するソフトウェア割込みは、該実行中の割込み処理より優先順位を低くする必要がある。事務処理OS割込みに対する割込み処理実行では、何らリアルタイムOSに影響を与えないためタスク切替判定255は必要としないため、このまま割込みハンドラ123の処理を終了する。このため、割込みスタック153上に退避しておいたレジスタ値を復帰し(処理259)、共通割込みハンドラへ制御を戻す(処理260)。なお、処理255を省くのは処理高速化のためであり、処理255を実行するように動作しても何ら問題はない。

【0050】一方リアルタイムOS割込みに対応する割込み処理実行後(処理253)、タスク切替えが必要か否かを判定する(処理255)。タスク切替えが必要なければ、割込みハンドラ123の処理を終了する。このため、割込みスタック151上に退避しておいたレジスタ値を復帰し(処理259)、共通割込みハンドラへ制御を戻す(処理260)。タスク切替えが必要であると判定した場合、割込みスタック151上に退避したレジスタの値をタスク管理テーブルにコピーし(処理256)、割込みスタック上の退避レジスタを破棄する(処理257)。この後、リスケジューラ119を起動する(処理258)。なお、プロセッサ100に、メモリ101上データ参照と同時に割込みスタックポインタ152の値を増減できる機能が備わっていれば、処理256と処理257を纏めて実行することができる。

【0051】一方、割込みハンドラ123で生成したソフトウェア割込みにより起動される事務処理OSの割込みハンドラ122は、その大部分を図10で説明した動作フローと同内容で処理を実行する。なお、一部動作が異なるのが処理181に示した割込みに対応する割込み処理の実行である。本動作例では、図16の処理254において事務処理OS割込みに対する割込み処理を実行し、ここで入手した情報は、共有メモリ125に書き込まれているものとする。従って、割込みハンドラ122内の処理181は共有メモリ125に記録された前記情報を読み出し、割込みに対応する処理を実行するように動作することで、直接入出力装置をアクセスすることなく割込み処理が実現される。

【0052】また、図14で説明したようにクライアントサーバーモデルを適用し複数のオペレーティングシ

テムで入出力装置を共用する場合においても、図11～図13の説明同様、割込みの内容を診断しないといずれのオペレーティングシステムの割込み処理を実行してよいか判断できないケースが発生する。本ケースでのリアルタイムOS117に備えられた割込みハンドラ123の処理フローを図17を用い説明する。なお、割込み内容を判定するためのテーブルは前記説明同様にスイッチテーブル200を用いた場合を想定するが、ここで説明する処理はスイッチのみならず、他の入出力装置に適用することも可能である。

【0053】第一に、使用中レジスタを割込みスタック153に退避し(処理270)、優先先割込みテーブル156を参照することで割込み要求した入出力装置が複数のオペレーティングシステムから共有されているか判定する(処理271)。共有していない場合は、リアルタイムOSに対する割込み要求であり、対応する割込み処理を実行する(処理274)。共有フラグが立っている場合は、割込み内容を判定することで割込み対象オペレーティングシステムを判定する(処理272)。割込み内容とスイッチテーブル200を比較し、リアルタイムOS専用スイッチが押されたと判定した場合はリアルタイムOS117を、事務処理OS専用スイッチが押されたと判定した場合は事務処理OS116を割込み対象OSとしてそれぞれ記憶する。また共用スイッチが押されたと判定した場合は、優先割込みテーブル156を参照し、フラグが立っているOSを割込み対象OSとして記憶する。図7に示す優先割込みテーブル156と、図12のスイッチテーブル200を用い、かつスイッチ割込みにIRQ#2が割り当てられている場合を想定すると、スイッチ#0が入力された場合はリアルタイムOS、スイッチ#2が入力された場合は事務処理OS、スイッチ#3が入力された場合は共用なのでさらに優先割込みテーブル156を参照しリアルタイムOSというように対応する割込み対象オペレーティングシステムを入手することができる。

【0054】次に割込み対象OSを参照し(処理273)、リアルタイムOS117であればリアルタイムOS割込みに対する割込み処理を実行する(処理274)。事務処理OS116であれば事務処理OS割込みに対する割込み処理274をそれぞれ実行する(処理275)。ここで、なお、事務処理OS割込みに対する割込み処理275は、割込みが発生したことをクライアント側の割込みハンドラ122に知らせるため、事務処理OS116へのソフトウェア割込みを生成する。さらに必要があれば前記ソフトウェア割込み処理で入手した情報を、共有メモリ125に書き込む。なお生成するソフトウェア割込みは、該実行中の割込み処理より優先順位を低くする必要がある。

【0055】事務処理OS割込みに対応する割込み処理実行では、何らリアルタイムOSに影響を与えないため

タスク切替判定276は必要としない。従って、このまま割込みハンドラ123の処理を終了する。そこで、割込みスタック153上に退避しておいたレジスタ値を復帰し(処理280)、共通割込みハンドラへ制御を戻す(処理281)。なお、処理276を省くのは処理高速化のためであり、処理276を実行するように動作しても何ら問題はない。

【0056】一方リアルタイムOS割込みに対応する割込み処理実行後、タスク切替が必要か否かを判定する(処理276)。タスク切替が必要なければ、割込みハンドラ123の処理を終了する。このため、割込みスタック153上に退避しておいたレジスタ値を復帰し(処理280)、共通割込みハンドラへ制御を戻す(処理281)。タスク切替が必要であると判定した場合、割込みスタック151上に退避したレジスタの値をタスク管理テーブルにコピーし(処理277)、割込みスタック上の退避レジスタを破棄する(処理278)。この後、リスケジューラ119を起動する(処理279)。

【0057】以上、複数のオペレーティングシステムから同一の入出力装置を共有するための方法としてクライアントサーバモデルを適用する方法について説明した。この中でサーバ側のオペレーティングシステムから他方のオペレーティングシステムへ割込み発生をソフトウェア割込みで知らせる方法につき説明したが、何らかの通信手段を備えていれば割込み発生をサーバ側入出力処理222からクライアント側入出力処理221へ通知することが可能であり、前記説明したソフトウェア割込みと置き換えることができる。本実施例では、2つのオペレーティングシステムの間で通信するため、図1に示すようにOS間通信機能126を備える。そこでOS間通信機能126を用い、複数のオペレーティングシステムから同一の入出力装置を共有するための方法を図18を用い説明する。

【0058】図18における構成要素は、その大部分が図1および図5で説明した構成要素と同一であるが、事務処理OS116に備えられていた入出力ドライバ120、割込みハンドラ122に代えて仮想入出力ドライバ290、仮想割込みハンドラ291を備えたところが異なる。仮想入出力ドライバ290は、アプリケーションタスクから入出力装置をアクセスするためのインタフェースを提供するが、直接入出力装置に読み書きしたり、直接入出力装置を制御することはしない。ここでは、OS間通信機能126が提供するメッセージ通信を用い、入出力ドライバ121に入出力装置への読み書きや制御を依頼するとともに、その結果もメッセージ通信を用い入出力ドライバ121から受け取るように動作する。仮想割込みハンドラ291は、割込みハンドラ123から送られるメッセージで起動され、割込み処理を実行する。

【0059】次に、双方のオペレーティングシステムが共有する入出力装置割込み要求が発生したときの処理の流れを図18を用い説明する。説明の都合上、ここで扱う入出力装置が生成する割込み番号の優先割込みテーブルには、共有フラグ1がセットされ、さらに事務処理OS側にフラグが立っているものとする。

【0060】入出力装置で発生した割込み要求は、一旦共通割込みハンドラ128にトラップされ、図15のフローに従って処理を実行する。ここでは両方のオペレーティングシステムが共有する入出力装置からの割込みであると仮定しているため、リアルタイムOSに備えられた割込みハンドラ123を起動する。割込みハンドラ123は、図16で説明したフローに従って処理を実行する。ここでは、優先割込みテーブル156に共有フラグ、及び事務処理OSにフラグが立っていると仮定しているため、事務処理OS割込みに対応する割込み処理254が実行される。ここで、事務処理OS割込みに対する割込み処理254は、割込みが発生したことをクライアント側の仮想割込みハンドラ291に知らせるため、OS間通信機能126を用い仮想割込みハンドラ291へメッセージを送る。なお、割込み処理で入手した情報は、メッセージに添付して送信してもよいし、共有メモリ125に書き込み仮想割込みハンドラ291から参照することで情報転送してもよい。さらに、仮想割込みハンドラ291にメッセージを受信するためのメッセージキューを設けられない時は、アプリケーションタスクとして備えられたサーバタスク112に一旦メッセージを送り、該サーバタスク112から仮想割込みハンドラ291を呼び出すことで仮想割込みハンドラ291を起動してもよい。仮想割込みハンドラ291は、メッセージに添付された割込み処理の処理結果、ないし共有メモリ125を参照することで割込み処理を実行される。

【0061】さらに、事務処理OSが備えるタスクから依頼された入出力処理を実行する処理の流れを図19を用い説明する。事務処理OS内の仮想入出力ドライバ290は、事務処理OSから入出力装置をアクセスするための唯一のインタフェースであり、依頼された入出力処理はOS間通信機能126を用い入出力ドライバ121へ転送される。なお、仮想入出力ドライバ290で入手した情報は、メッセージに添付して送信してもよいし、共有メモリ125に書き込み入出力ドライバ121から参照することで情報転送してもよい。さらに、入出力ドライバ121にメッセージを受信するためのメッセージキューを設けられない時は、アプリケーションタスクとして備えられたサーバタスク113に一旦メッセージを送り、サーバタスク113から入出力ドライバ121を呼び出すことで入出力ドライバ121を起動してもよい。入出力ドライバ121は、メッセージに添付された依頼内容、ないし共有メモリ125を参照することで入出力処理を実行する。

【0062】次に、OS間制御機能124に備えられた優先割込みテーブル156の書き換えフローを図2-0を用い説明する。優先割込みテーブル156は、ユーザ要求を管理する管理タスク115等によって書き換える。しかし、優先割込みテーブル156を書き換えている最中に、割込み処理が優先割込みテーブル156を参照すると不整合が発生するおそれがある。そこで最初に全割込み要求をマスクし、割込みが発生しないようにする

(処理301)。なお書き換えを実施する割込み番号のみ割込み要求をマスクしてもよい。割込みマスクの設定後、優先割込みテーブル156の内容を更新し(処理302)、設定した割込みマスクを解除する(処理303)ことで優先割込みテーブル156の書き換えが実施できる。

【0063】以上、複数のオペレーティングシステムから入出力装置を共有する方法を説明したが、実際の応用例として車載ナビゲーション装置といったユーザインタフェースを備える組み込み装置のグラフィックス表示に本発明を適用した例を図21を用い説明する。なお、グラフィックス表示システム上で動作するオペレーティングシステムは、図1同様に事務処理OS116、及びリアルタイムOS117で、リアルタイムOS上で動作するタスク/スレッドは事務処理OS上で動作するタスク/スレッドより優先順位が高いと仮定する。

【0064】グラフィックス表示装置は、描画や表示といったコマンドを解釈しグラフィックスハードウェア314を制御するグラフィックスドライバ310、312、グラフィックスハードウェア314が生成する割込み要求で起動するグラフィックス割込みハンドラ311、313、及び画面に展開されたイメージデータをディスプレイに表示するグラフィックスハードウェア314で構成する。グラフィックスハードウェア314は、各画面の輝度値を保持するフレームメモリ316、及び垂直同期周波数毎にフレームメモリ316から各画面の輝度値を取り出し、ディスプレイ104に出力する表示制御315とを備える。ここで、フレームメモリ316では、事務処理OSのグラフィックスドライバ310が作成したイメージを蓄える事務処理OS用描画領域317と、リアルタイムOSのグラフィックスドライバ312が作成したイメージを蓄えるリアルタイムOS用描画領域318とを独立・分離することで、リソースの衝突を防ぐ。これにより、オペレーティングシステムを切り替えた場合でも、各々の描画イメージは保持されるため、再度オペレーティングシステムが切り替わったときでも、切り替わり前の状態から処理を続行できるという利点を有する。また、リアルタイムOSに備えられたグラフィックスドライバ312、及びグラフィックス割込みハンドラ313はサーバとして動作し、事務処理OSに備えられたグラフィックスドライバ310、及びグラフィックス割込みハンドラ311はクライアントとして動

作すると仮定する。

【0065】リアルタイムOS117に組み込まれたアプリケーションタスクからの描画、表示要求における動作フローを最初に説明する。アプリケーションタスクは、グラフィックスドライバ312に備えられた描画コマンドを呼び出し、描画処理を実行する。描画コマンドは、線や多角形の描画するためのコマンドや線幅や塗りつぶしパターンといった属性を指定するコマンドで構成され、グラフィックスドライバ312は描画コマンドを解釈し、リアルタイムOS用描画領域318に画素展開する。グラフィックスハードウェア314が、画素展開をハードウェアで実行するアクセラレーション機能を有していれば、それを活用してもよい。描画が終了すると、次にグラフィックスドライバ312に備えられた表示コマンドを実行する。表示コマンドは、フレームメモリ316における表示開始アドレスや画面サイズを指定するコマンドで構成され、表示制御315内のレジスタを制御することで指定された画像を表示する。指定された画面の表示が開始されると、表示制御315は割込み要求を生成する。割込み要求は、一旦共通割込みハンドラ128にトラップされるが、直ちに共通デバイスからの割込み要求と判定し、グラフィックス割込みハンドラ313を呼び出す。これによりグラフィックス割込みハンドラ313は、画面が表示されたことを識別する。

【0066】次に、事務処理OS116に組み込まれたアプリケーションタスクからの描画、表示要求における動作フローを次に説明する。アプリケーションタスクは、グラフィックスドライバ310に備えられた描画コマンドを呼び出し、事務処理OS用描画領域317に画素展開する。描画が終了すると、次にグラフィックスドライバ310に備えられた表示コマンドを実行する。表示コマンドは、フレームメモリ316における表示開始アドレスや画面サイズを指定するメッセージをOS間通信機能126を用いグラフィックスドライバ312に転送する。その要求を受けたグラフィックスドライバ312は、表示制御315内のレジスタを制御することで指定された画像を表示する。なお、グラフィックスドライバ312は、いずれのオペレーティングシステムからの表示要求が優先であるかを判定し、常に優先的に表示すべきフレームメモリを表示する。指定された画面の表示が開始されると、表示制御315は割込み要求を生成する。割込み要求は、一旦共通割込みハンドラ128でトラップされるが、直ちに共通デバイスからの割込み要求と判定し、グラフィックス割込みハンドラ313を起動する。グラフィックス割込みハンドラ313は、OS間通信機能126を用い、割込み要求を受けたことをグラフィックス割込みハンドラ311に伝える。これによりグラフィックス割込みハンドラ311は、画面が表示されたことを識別する。

【0067】以上、本発明を適用したグラフィックス表

示装置の具体的な実施例を説明したが、本発明を適用することで例えばシリアル／ネットワーク通信やCD-ROM/DVD-ROM記憶装置といった様々な周辺装置を複数のオペレーティングシステムから共有することが可能になる。

#### 【0068】

【発明の効果】本発明によれば、複数個のオペレーティングシステムを単一のプロセッサで動作させる計算機システムにおいて、各オペレーティングシステムが実行するアプリケーションタスクから同一の入出力装置を共有することが可能になるため、入出力装置の総数を削減することが可能になる。さらに、本発明によれば、複数個のオペレーティングシステムが共有する入出力装置とデータ入出力するための入出力ドライバ、及び割込みハンドラを該一つのオペレーティングシステムに実装し、それ以外のオペレーティングシステムから該入出力ドライバ及び割込みハンドラに入出力処理を依頼することで、入出力装置をリセットすることなく入出力処理を継続することが可能になり、操作性が向上する。

#### 【図面の簡単な説明】

【図1】本発明の第1の実施例におけるシステム構成を示す図である。

【図2】計算機システムのハードウェア構成を示す図である。

【図3】割込みレベルマスク機能を装備する場合のステータスレジスタを示す図である。

【図4】個別割込みマスク機能を装備する場合のステータスレジスタを示す図である。

【図5】割込み処理プログラムの詳細を示す第1の図である。

【図6】割込みアドレステーブルの構成を示す図である。

【図7】優先割込みテーブルの構成を示す図である。

【図8】OSコンテキスト切換モジュールの処理フローを示す図である。

【図9】共通割込みハンドラの処理フローを示す第1の図である。

【図10】割込みハンドラの処理フローを示す第1の図である。

【図11】車載ナビゲーション装置の入出力パネルの構成を示す図である。

【図12】スイッチテーブルの構成を示す図である。

【図13】共通割込みハンドラの処理フローを示す第2の図である。

【図14】本発明の第2の実施例におけるシステム構成を示す図である。

【図15】共通割込みハンドラの処理フローを示す第3

の図である。

【図16】割込みハンドラの処理フローを示す第2の図である。

【図17】割込みハンドラの処理フローを示す第3の図である。

【図18】入出力処理プログラムの詳細を示す第1の図である。

【図19】入出力処理プログラムの詳細を示す第2の図である。

【図20】優先割込みテーブル書き換えフローを示す図である。

【図21】本発明をグラフィックス表示装置に適用したシステムの構成を示す図である。

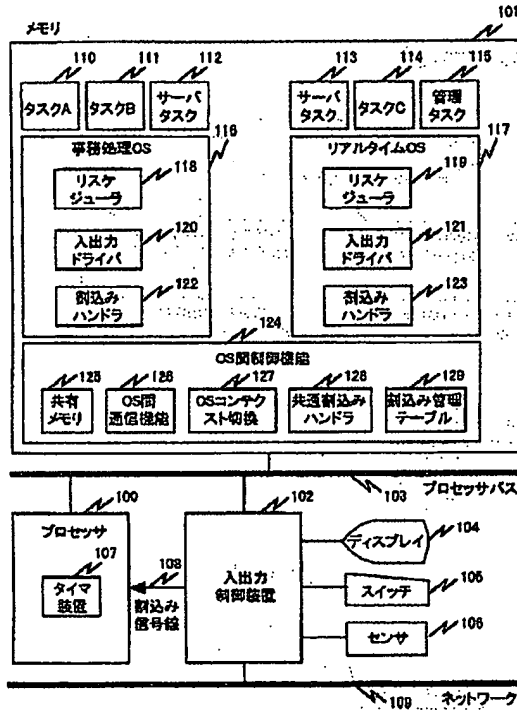
#### 【符号の説明】

100…プロセッサ、101…メモリ、102…入出力制御装置、103…プロセッサバス、104…ディスプレイ、105…スイッチ、106…センサ、107…タイマ装置、108…割込み信号線、109…ネットワーク、110…タスクA、111…タスクB、112、113…サーバタスク、114…タスクC、115…管理タスク、116…事務処理OS、117…リアルタイムOS、118、119…リスケジューラ、120、121…入出力ドライバ、122、123…割込みハンドラ、124…OS間制御機能、125…共有メモリ、126…OS間通信機能、127…OSコンテキスト切換、128…共通割込みハンドラ、129…割込み管理テーブル、130…キャッシュメモリ、131…CPU、132…汎用レジスタ、133…PC、134…SR、135…データバス、136…アドレスバス、137…割込みコントローラ、140…割込みブロックビット、141…割込みマスクレベル、142…実行状態レジスタ、143…割込みマスクレジスタ、144～147…割込みマスクビット、150、152…割込みスタックポインタ、151、153…割込みスタック、154…実行OS記憶、155…割込みアドレステーブル、156…優先割込みテーブル、157…事務処理OS用保存コンテキスト、158…リアルタイムOS用保存コンテキスト、190…リアルタイムOS専用スイッチ、191…事務処理OS専用スイッチ、192…共用スイッチ、200…スイッチテーブル、221…クライアント側入出力処理、222…サーバ側入出力処理、226…入出力装置、290…仮想入出力ドライバ、291…仮想割込みハンドラ、310、312…グラフィックスドライバ、311、313…グラフィックス割込みハンドラ、314…グラフィックスハードウェア、315…表示制御、316…フレームメモリ、317…事務処理OS用描画領域、318…リアルタイムOS用描画領域。



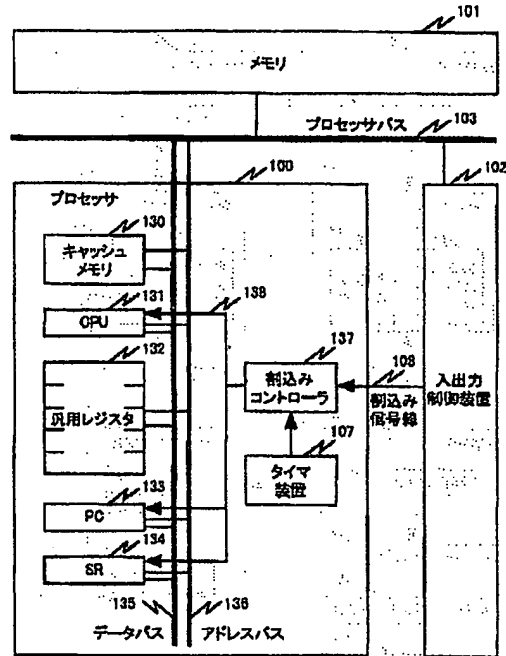
【図1】

図 1



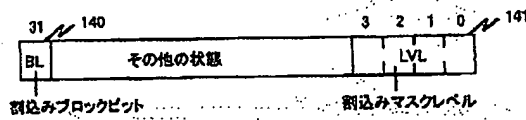
【図2】

図 2



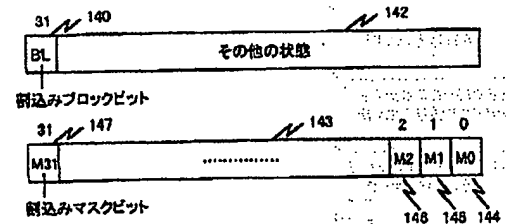
【図3】

図 3



【図4】

図 4



【図6】

図 6

割込みアドレステーブル 155

割込み番号	事務処理OS	リアルタイムOS
IRQ 0	0x0000400	0x00000600
IRQ 1	0x0000420	0x00000620
IRQ 2	0x0000440	0x00000640
IRQ 3	0x0000460	0x00000660
IRQ 4	0x0000480	0x00000680
...	...	...
IRQ 15	0x00005C0	0x000007C0

【図7】

図 7

優先割込みテーブル 156

割込み番号	事務処理OS	リアルタイムOS	共有
IRQ 0	0	1	0
IRQ 1	1	0	0
IRQ 2	0	1	1
IRQ 3	1	0	1
IRQ 4	0	1	0
...	...	...	...
IRQ 15	0	1	0

【図12】

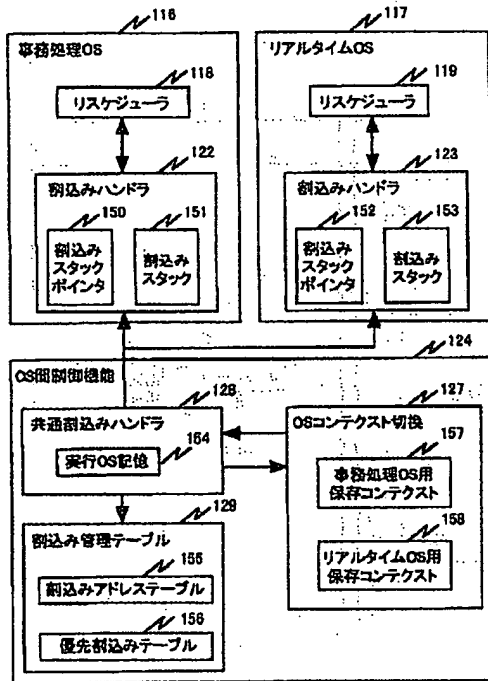
図 12

スイッチテーブル 200

スイッチ番号	属性
# 0	リアルタイムOS
# 1	リアルタイムOS
# 2	事務処理OS
# 3	共有
# 4	共有
...	...
# n	共有

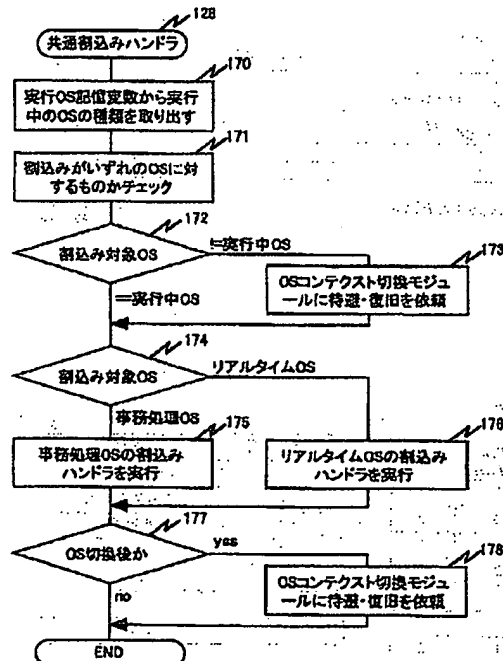
【図5】

図 5



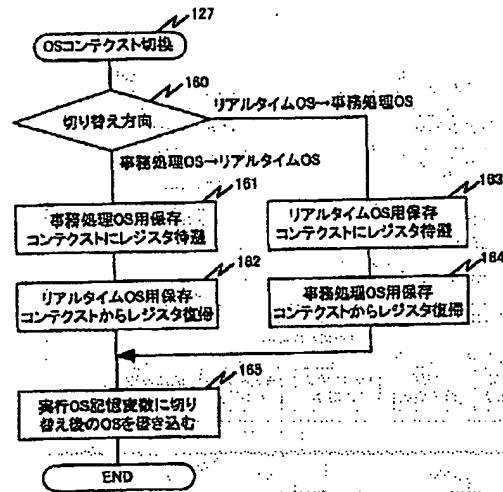
【図9】

図 9



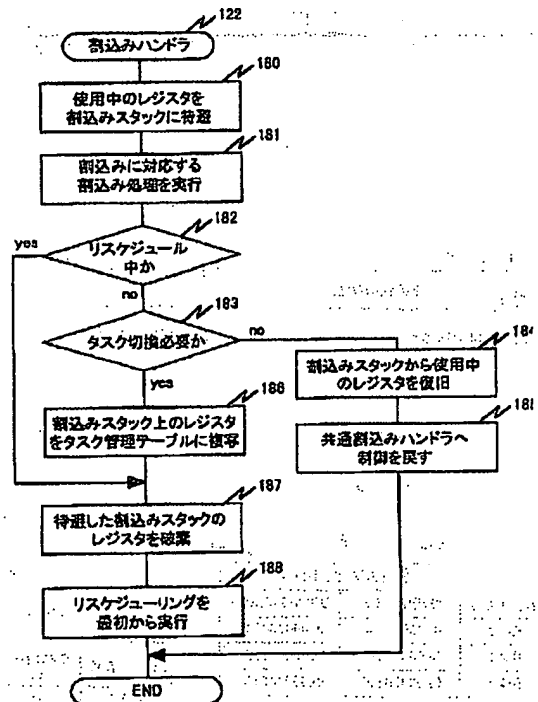
【図8】

図 8

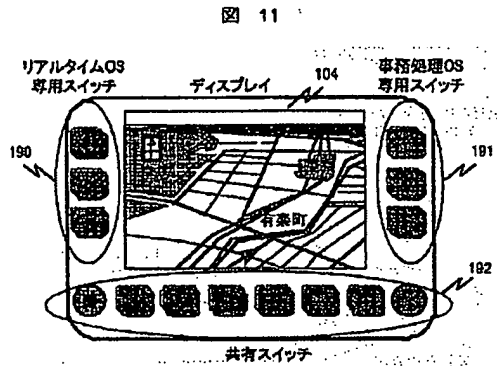


【図10】

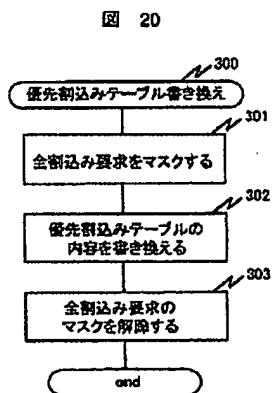
図 10



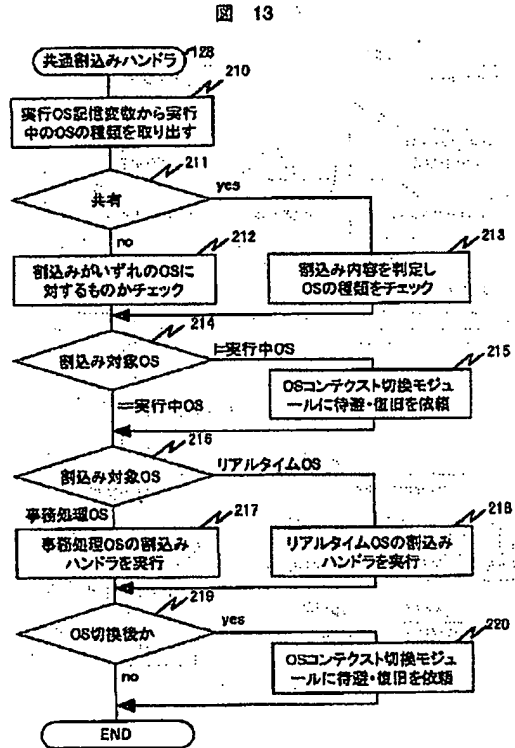
【図11】



【図20】

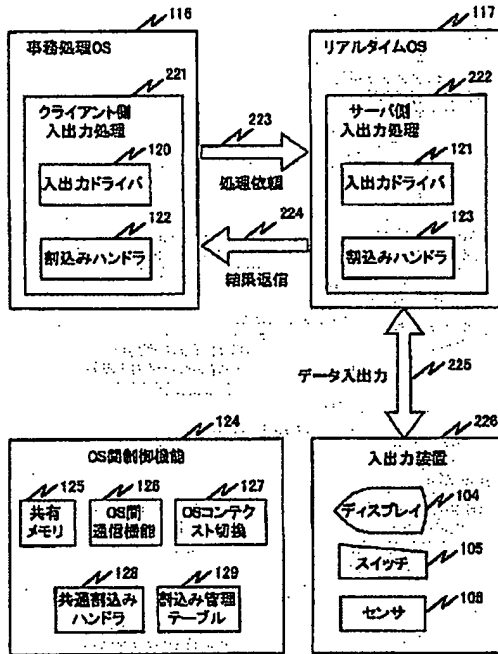


【図13】



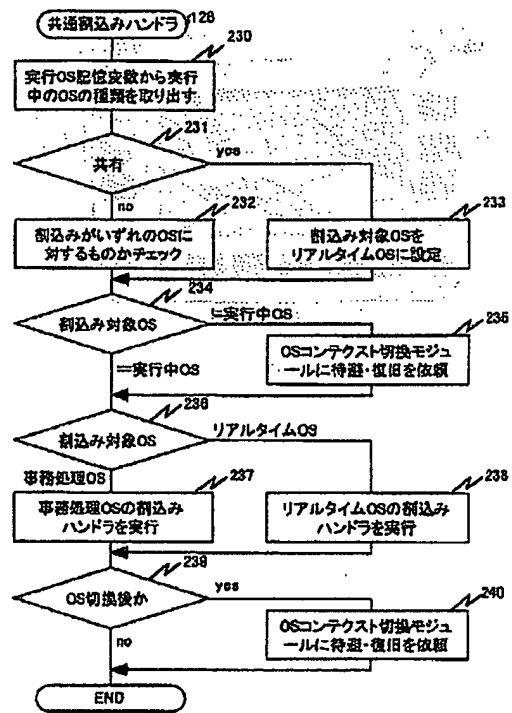
【図14】

図 14

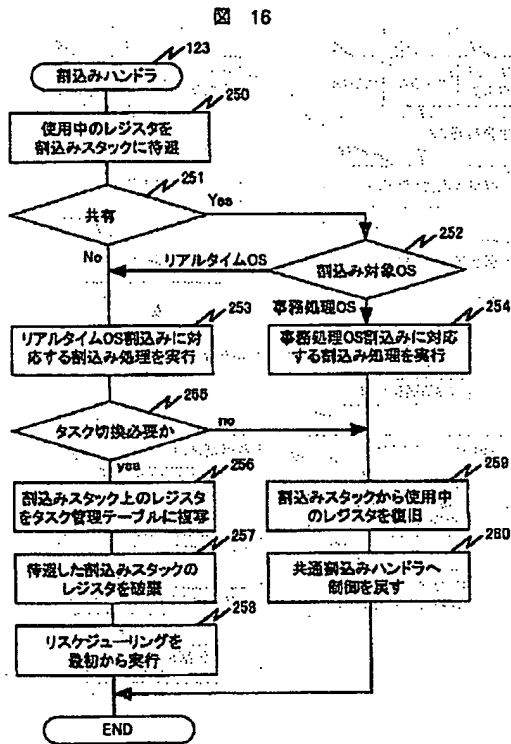


【図15】

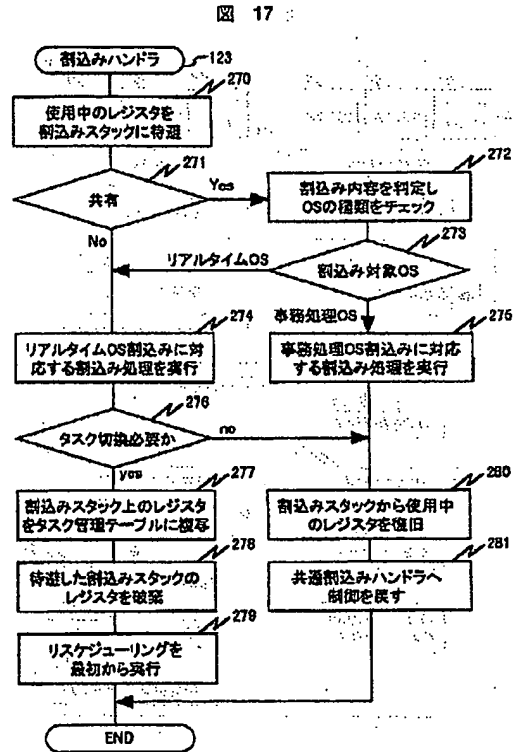
図 15



【図16】

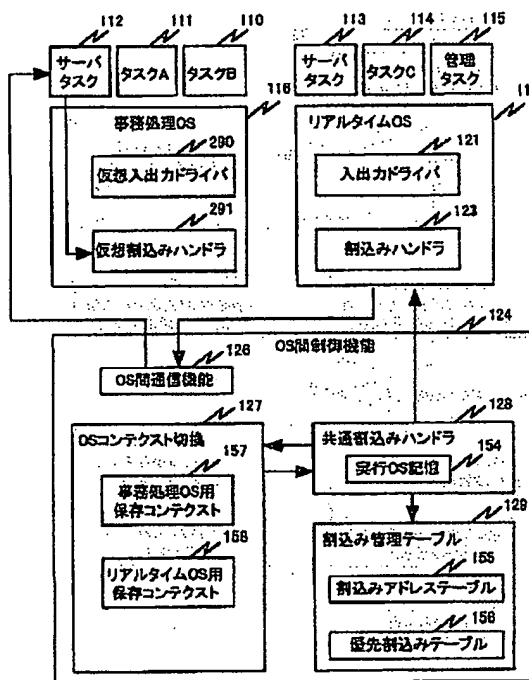


【図17】



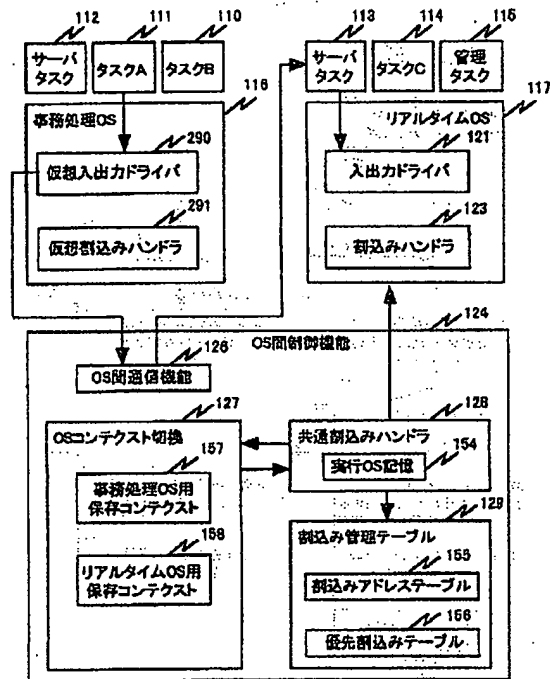
【図18】

図 18



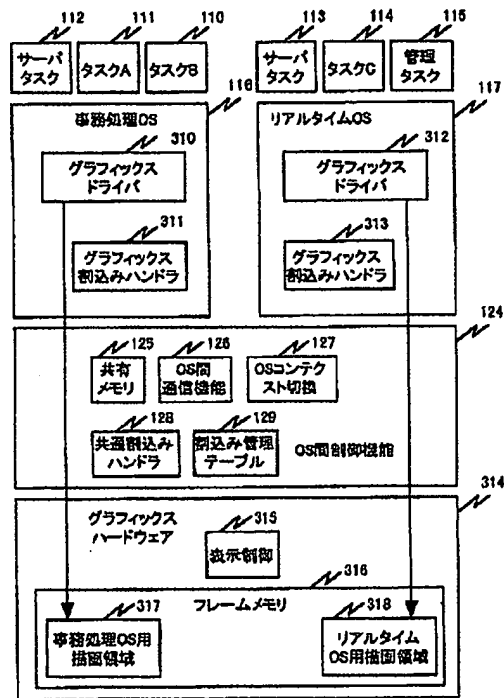
【図19】

図 19



【図21】

図 21



フロントページの続き

(72)発明者 上脇 正

茨城県日立市大みか町七丁目1番1号 株式会社日立製作所日立研究所内

(72)発明者 齋藤 雅彦

茨城県日立市大みか町七丁目1番1号 株式会社日立製作所日立研究所内

(72)発明者 大野 洋

東京都千代田区神田駿河台四丁目6番地 株式会社日立製作所新事業推進本部内

(72)発明者 中村 智明

東京都千代田区神田駿河台四丁目6番地 株式会社日立製作所新事業推進本部内

Fターム(参考) 5B014 FB04

5B076 AA14 AA15

5B098 AA05 BB11 GA02 GA04 GA08

GB01 GC03 GC16